Introduction to React

React Workshop - P2

What is React ?

- A JavaScript library for building user interfaces
- Open sourced to the world by Facebook and Instagram team in 2013
- React can be used as a base in the development of web or mobile applications (React Native)
- React is a small library that doesn’t come with everything you might need out of the box to build your application

React Elements

- Smallest building blocks of React apps
- An element describes what you want to see on the screen
- Unlike browser DOM elements, React elements are plain objects, and are cheap to create.
- React DOM takes care of updating the DOM to match the React elements – Rendering Process

```
const element = React.createElement("h1", null, "Hello, World!");
ReactDOM.render(element, document.getElementById('root'));
```

React Components

- A user interface is made up of parts, In React, we describe each of these parts as a component
- Components allow us to reuse the same structure and then we can populate those structures with different sets of data.
- Look for opportunities to break down your applications into reusable components

Defining React Components

```
// Function Component
function Welcome(props) {
  const message = "Hello, {props.name}";
  return React.createElement("h1", null, message);
}


// Class Component
class Welcome extends React.Component {
  render() {
    const message = "Hello, {props.name}";
    return React.createElement("h1", null, message);
  }
}
```

JSX

- A syntax extension to JavaScript
- A visual aid when working with UI inside the JavaScript code
- Provides a concise, readable and HTML like declarative syntax for creating complex UI
- JSX might look familiar, and most of the syntax is similar to HTML
- JSX produces React “elements”

JSX Tips

- CSS Class Attribute
  - Since class is a reserved word in JavaScript, className is used to define the class attribute
- JavaScript Expressions
  - JavaScript expressions are wrapped in curly braces and indicate where variables shall be evaluated and their resulting values returned
  - Values of types other than string should also be put as JavaScript expressions
- JSX can't be interpreted with a browser. All JSX must be converted into createElement calls
  - Tool for this task: Babel

JSX Represents Objects

```jsx
// JSX
const element = <h1 className="greeting">Hello, world!</h1>;

// createElement call
const element = React.createElement(
  "h1",
  { className: "greeting" },
  "Hello, world!"
);

// React internally create following object
const element = {
  type: "h1",
  props: {
    className: "greeting",
    children: "Hello, world!",
  },
};
```

Babel

- The first version of the project was called 6to5
- 6to5 was a tool that could be used to convert ES6 syntax to ES5 syntax, which was more widely supported by web browsers
- As the project grew, it aimed to be a platform to support all of the latest changes in ECMAScript
- It also grew to support converting JSX into JavaScript
- The project was renamed as Babel in February 2015

React Fragments

```
render() {
  return (
    <React.Fragment>
      <ChildA />
      <ChildB />
      <ChildC />
    </React.Fragment>
  );
}
```

- React will not render two or more adjacent or sibling elements as a component
- We used to have to wrap these in an enclosing tag like a div.
- This led to a lot of unnecessary tags being created though, a bunch of wrappers without much purpose.
- If we use a React Fragment, we can mimic the behaviour of a wrapper without actually creating a new tag

Webpack and 'create-react-app'

- Webpack is a module bundler
- A module bundler takes all of our different files (JavaScript, JSX, ESNext) and turns them into a single file
- The two main benefits of bundling are modularity and network performance
- 'create-react-app'
  - Help developers get started with React projects quickly without the manual configuration of webpack